Adventure CODER

AC 2023

WRITE TO...

CHRISTOPHER HESTER
17 HILL FOOT
SHIPLEY
WEST YORKSHIRE
BD18 4EP

SUBSCRIPTIONS

Each issue costs...

£1.50	Britain, Isle Of Man, Channel Islands
£1.90	Europe, Republic Of Ireland
£2.35	Zone 1 (Canada, USA, Falklands, Seychelles, Saudi Arabia, etc.)
£2.45	Zone 2 (New Zealand, Australia, Japan, Fiji, Philippines, etc.)

Airmail applies outside Britain.

PAYMENT

Cheques are payable to
"C.HESTER" only. Or you can
use Postal Orders, International
Giros, or British decimal stamps
for small amounts. (No coins!)

LETTERS

If you don't want your letter to be
printed, put "Not for publication"
on the top. If you would like a
personal reply, please enclose a
stamped addressed envelope.

CONTRIBUTIONS

These are always welcome, and keep
the magazine going! If you can, PRINT
your text on white A4 paper in dark ink,
with a margin of about an inch around
each edge. Handwritten text is also
welcome! As is text from a computer,
saved onto a 1½" floppy disk, in the
ASCII format. All disks returned.

ABOUT THIS MAGAZINE

Adventure Coder is an amateur,
non-profit publication appearing on an
irregular basis. Items credited to an
author (and not the Editor) remain
the copyright of that author, who is
free to use the same item elsewhere.
All other items\pages © C. Hester 1995.

PLEASE NOTE

The views expressed within this
magazine may not correspond to
those of the Editor, C. Hester.

ADVERTISING RATES

Full page - £5.00
Half page - £2.50
Quarter page - £1.25

Line ads are also accepted FREE.
However, the Editor retains the
right to refuse to print line ads
liable to be potentially damaging,
illegal, or in bad taste.

Version 1.0   June 1995

CONTENTS

Cartoon by Ev.

Editorial

Natural Born Coders

Welcome to Issue 23 of *Adventure Coder!* I hope you agree that the wait was worth it! I know it was last year when the previous issue came out, but that's nothing compared to the wait fans of The Stone Roses have had to endure between the release of their debut album and their second album - a mere *five* years... so at least I didn't take that long!

Looking inside this issue, you can't fail to notice the improvements I've made in the design and layout. This is what's taken so long - and there's a lot more to do yet. You see, it's like this... I finally gave in to desire, and splashed out on a new PC computer! I went for a basic one, leaving out the CD-and-speakers Multimedia option, and so was able to afford it - the final cost was well under a thousand pounds - why pay more for things you don't really need? (You can add a Multimedia set-up later, when money allows!) That's the great thing about PCs, and the bad thing about the earlier computers, like my previous Atari ST. You can upgrade your PC almost ad infinitum, whereas the ST had limits - it's now a dying machine. The real nail in its coffin came when I read that it'd need its memory doubling and a hard drive fitting, before it could be used properly to access the phone-based internet computer system - and then without you being able to use a key program (*Mosaic*), and then, and then.... bye bye ST! Now with my new PC, although it only has a basic "DX" processor chip (running at 33MHz - that's slow thesedays!), I can replace this at any time with a "DX2", "DX4" or "Pentium" chip, and gain instant speed improvements. The odd thing is, I find the PC that fast anyway, I don't yet feel the need to upgrade it!

I've also now got a built-in hard drive, which is brilliant for storing masses of programs and files. The graphics are excellent too, with a whopping 16 million colours - I could never go back to my measly ST now! The PC ain't perfect, but if anyone hasn't used one yet, I'd urge you to do so immediately!! Not for the hardware, but for the fabulous software - I refer, of course, to the mighty (if ageing) *Windows!* This sets up a graphical environment where things are manipulated on the screen, in a really simple and effective way - no horrible text commands to get just right. I found a task I needed to do, such as copying a file, was obvious just by the way it all works. And the best part is, nearly all programs work alongside *Windows*, so they share similar ideas - you don't need the manuals! This is also down to the inbuilt Help menu available - each program has its own screens of help-text, which are layered, so you can keep selecting help on something until you find the exact bit of help you need - absolutely fantastic! It makes computers so much more fun!

Naturally, I've been using the PC to do some of this issue - see if you can spot the pages! I've even been able to use some of my text files saved from the ST, just by shoving the ST disk into the PC's disk drive - it managed to read the files! (I had to alter them to look right, but the text was there! That saved a lot of retyping!) One irresistible aspect of PCs is the use of "True-Type" fonts which you get. These enable a wide range of styles for any text you create - such as the font I've used above for the "Editorial" text. That's a font called *Bailey*, and I've got hundreds of other great-looking fonts installed, after buying some smart new font-packs. The point about these is that you can enlarge them and reduce them with virtually no loss of quality (unlike normal computer images which get very blocky if enlarged) as they are all made using "vector" data - they're drawn on-the-fly, using sharp edges every time. The result is sheer professionalism! You could get these fonts on an ST, but not without paying a lot, I found. The PC has so many fonts available for it, you'd need a massive amount of memory just to use them all! (My PC's got a standard 4Mb of memory by the way - it's a bit tricky for some newer programs, which would prefer 8Mb, the new standard, yet it's still four times the amount I had in my ST!)

Oh - I nearly forgot. When you're running a program such as a word processor, you can keep this in memory if you like, and load in a paint program as well. And as many other programs as your PC's memory will allow!! With the ST, and other older machines, it was often a case of one program only. But that's not all. You can keep switching between the programs at will, as they remain just as you left them! Not impressed yet? Well how about this - you can, thanks to *Windows*, take data from one program - and paste it into another! Graphics can even be culled from a paint program and pasted straight into a word processor!! (And text can likewise be taken from that, and pasted onto a picture, whatever.) All while the programs are in the memory at once! (And that's just the tip of a huge iceberg of useful little things that are possible - way too many to go into here!) Another thing I only realised recently, is that you can even load in the same program several times!!! Each version can be different - for instance your word processor could be loaded in firstly with a letter, then again with an article, a third time with a list. all versions able to be accessed and worked on! (Without having to stop and save the file, load the next one, work on it, save it, load the old file back again, yawn, yawn.) It's a truly user-friendly system!

Your Editor, *Chris.*

GET REAL

by Steve Clay

Fact, as the saying goes, can be stranger than fiction. Here are few pieces of information that could be adapted for use within an adventure.

The Archerfish.
This nifty little chap spits a jet of water at flies skimming the surface above. The fly is stunned, drops onto the water and the Archerfish has lunch. Increase the size of the fish and you could have the cause of the Bermuda triangle. The players task could be to destroy this pest.

Elephants.
As everyone knows, the elephant will head for an elephants' graveyard when it is about to die. If the elephant was a swarm of mutant thingies heading for their spawning ground and the players' village was in direct line, then perhaps a game or scenario could be dreamt up. Maybe the Thingies need to consume human flesh before the spawning.

Whales.
Or rather the eight foot long intestinal worms that inhabit some of these creatures. Idea, the parasite becomes detached from its host and needs to find a new one. Or perhaps the player can be sent underground to destroy a parasite that has attached itself to the foundations of a city and is currently feeding on the inhabitants.

Chameleon.
The famous hue-changing lizard. The old tale of the chameleon bursting into flames when placed on a tartan background is a myth. Fortunately adventure authors seldom worry about myths. Lead a giant chameleon into a tartan walled room and see it burst into flame. Or give the chameleon a vain streak and have it die of shame when it reaches the tartan room. These are only a few ideas. *[Perhaps this is a game for Tartan Software? Sigh... - Ed]* Look out for books on weird and wonderful creatures in the library.

AMOS Avenue

By John Ferris

Here is the second section of the program listing. As before, any text with "*****" attached is a comment, not part of the program.

```
DCODE:
FR = Free
Rem — Identify words as verbs nouns or adverbs—
U$ = Left$(W$(T),4)
If V < >0 Then Goto NO1
Z = 0
While Z < = VTOT and V = 0
  Inc Z : If U$ = VB$(Z) Then V = Z
Wend
If A = 2 Then Return
If V < >0 Then Return
```

****The Free statement forces garbage collection to tidy up the variable area and could be anywhere in the program as long as it is called reasonably frequently. If V (the verb number) is not 0, ie. a verb has already been found, the program jumps to NO1: else the verb array VB$() is searched to make a match. If A is 2 (see last listing, this means that a single letter command (eg. "N" or "S") has been typed by the player, this is either a verb or nothing) the routine ends. Finally, if a verb has been found, the routine ends.*****

```
NO1:
 If N(1) < >0 Then Goto NO2
 Z = 0
While Z < = NTOT and N(1) = 0
 Inc Z : If U$ = N$(Z) Then N(1) = Z
 Wend
If N(1) < >0 Then Return
```

****This finds the first noun in the command****

```
NO2:
If N(2) < >0 Then Goto ADVERB
 Z = 0
While Z < = NTOT and N(2) = 0
 Inc Z : If U$ = N$(Z) Then N(2) = Z
Wend
If N(2) < >0 Then Return
```
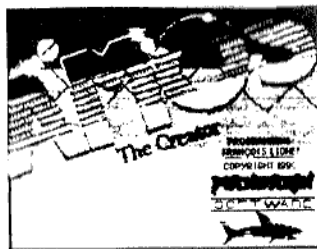
****The second noun, if any****
```
ADVERB:
If AD < >0 Then Return
Z = 0
While Z < = AOT and AD = 0
Inc Z : If U$ = AD$(Z) Then AD = Z
Wend
Return
```

****Finally the adverb is found, if it exists.****

ROOMPRINT:
Rem If LR=R Then Goto RRR
Rem Screen 3
Rem Load Iff RS$(R)
Screen 2
Rem If ST=0 Then Wind Open 2,1,89,78,17,1 : ST=1 : Pen 0 : Flash Off :
Rem Get Palette 3

****The above part of the ROOMPRINT: routine deals with loading graphics into screen 3, if
it is a graphical game. This assumes that there is a graphic for each room with the same
filename as the short description of the room. This could be changed to whatever system you
like. Additional arrays could be made to describe if a location has graphics and the name of
the graphics file to load. LR is the last room counter, if it is the same as the present room
then we haven't changed location so we don't need to reload the graphics. If ST=0 then we
are at the start of the game, so a text window is opened in screen 2, making the display
neater.****

RRR:
If R>1 and Z(2)=0 Then Goto DK

****In this game, Z(2)=1 if the crystal on the staff is glowing, else there is no light source and
all the locations greater than 1 are dark.****

If R>1
AVAIL[1] : If Param=False
Goto DK
End If : End If

**** Again, if object 1 is not AVAILable (Param=False) in a location greater than 1, then it is
dark. This caters for the player dropping the staff (still light) and moving into a dark location.
The staff is no longer AVAILable (not in hand or in the present location) to provide light.****

WRAP[R$(R)] : For T=1 To 7 : HERE[T] : If Param Then Q$=Q$+" "+OD$(T)+"," : Q=1
Next
If Q=1 Then Right$(Q$,1)="." : Print "Nearbye is ";Q$
Q$="" : Q=0
Rem Title Top RS$(R)

****WRAP is the wordwrap procedure, used in place of most Print commands in this
program. After the location description is printed, the 7 takeable objects tested to see whether
they are in the present location. HERE[T] returns Param=True is object T is lying about in
the location. If an object is present, its description (eg. "a staff") is added the the string Q$.
Note, if there are no objects in the location, nothing is printed at all, no "You can see nothing
else." message. Finally Q$ and Q are reset, and if the game contains graphics, the Rem'ed
line would write the short description of the present location into the text window border.
This is one reason why the short description is one word.****

If R=1 and EX(1,3)=2 Then WRAP[M$(3)] : Return
If R=6
If EX(6,1)=0 : WRAP[M$(21)] : Return : End If
If EX(6,1)=7 : WRAP[M$(22)] : Return : End If

7

End If
Return

***This part of ROOMPRINT deals with anything which might effect the room description itself, like the appearance of new exits. The first line detects if the secret door in the first location has been opened (the east exit leads to location 2) and so message three is printed saying that a door leads east. In room 6, if there's no north exit, then the door is still there, if there is, then say there's an exit in that direction.

DK: Print "IT'S TOO DARK TO SEE!": Return

****I'll give you ten guesses as to what this line does.****

```
HIP:
LR=R : TURN=TURN+1
If R=1
If TURN=8 : Print M$(7) : End If
End If
Rem this is for things happening after roomprint and before input
Return
```

****This is the high priority section and is called after ROOMPRINT: but before the player gets to type anything. HIP: is used for events that happen only once or do not affect the room description. HIP: is not called when the player types "LOOK". Non-player characters can be dealt with here, eg, a dog might be following the player around the game. HIP: would be used to set the location of the dog to the current location and print a "Fido is here" message.

The TURN variable is updated along with the LR (LastRoom number visited) variable. In location 1, there are only 7 moves of sunlight available for the staff to operate the secret door (via the crystal). On the eighth move, "The sun goes behind a cloud." is printed. The secret door is closed forever in the game. Note, the Print command is used because the message is less than 80 characters and Print is quicker than the WRAP: procedure.****

We are about a third of the way through the listing, what comes next is the CONDITIONS: routine which gets down to the hard work of responding to the players commands. You may have noticed the Z() array. I use this array as counters and flags. In this game, they have the following meanings:

$Z(1)$ =0 if staff is not in hole in loc.1, 1 if the staff is in the hole, =2 if the secret door is wedged open.
$Z(2)$ =0 if it is dark or 1 if the crystal on the staff is glowing.
$Z(3)$ =0 if door in loc.3 is locked, 1 if unlocked.
$Z(4)$ =0 if the seed packet is closed, 1 if it's open.
$Z(5)$ =1 if seed is in the hole in loc.1.
$Z(6)$ =1 when plant grown.
$Z(7)$ =0 when the jug is empty, 1 when the jug is full
$Z(8)$ =0 when the seed packet is full, 1 when it is empty.

Until next time.

The Shredder

Our new columnist takes a side-ways look at life, in the way only he can

Portable Blues

I'm writing this on my computer. Only it's not a PC (Personal Computer). Mine's an **IC!** (Impersonal Computer!) At least it is after trashing a file I'd been working on; I'd been writing this file for several months, on and off. I decided one day to really make it special. So I spent a good while working solidly on it. My pride and joy was nearly complete. I took a break, and turned the computer off. It's actually a portable computer (boast, boast) so when you turn it off, the text file you're working on remains open. When you turn the computer back on, the text is there again on the screen, just as it was when you left it. Great. That is unless you accidentally reset the machine by holding down two certain keys as you turn it back on. If your file wasn't saved, you've had it! El Bingo, no file. As if you haven't already guessed, this is what happened to me! I accidentally reset the machine turning it back on, and lost my precious file for good! (It was saved, but only in the internal memory, which means you lose the file if the computer's reset! Nice one!)

After three boxes of Kleenex, I was able to face the situation easier. But I found that not only my file had vanished - but the silly computer had also reset many of its settings. After a reset, you're back to the horrible default settings that were there initially, but if you're like me, were soon changed to better ones. Also, loads of small text shortcuts I'd set up were lost.

The bad part is this though: after several minutes experimenting (turning the computer on and off, resetting it again to see what it did and didn't do) I found the hideous truth that, *despite what the manual says*, you can actually reset the machine with not two keys, but ONE! (Held down as you switch it on.) Well, this is clearly what I must have done, as I remember touching the keyboard before, when turning the computer back on. (Only I didn't think I touched both of the keys it says are necessary to cause a reset.) This is clearly a lethal situation, as it's far too easy to reset the computer with just the one key. There's not even any warning message printed up, such as "Do you really want to reset this computer? Yes or No". Why not!? And why does the manual not warn users of the ability to reset the machine with only one key? Perhaps the programmers forgot to take out this 'feature' when they were testing the thing. Give me their address now, so I can start work on the hate mail!!

At the time, I was so annoyed, I felt like throwing the computer through the window! I couldn't face using it, for days after that experience. I've gone back to it now, of course - you can't keep me and computers apart for long, you know - but with an added sense of distrust. The feeling of fun I'd had before with this portable had been worn off for good. I still like it, and don't think about what happened all the time. Besides, who hasn't had a similar incident with their computer? If you've never lost any data yourself, then you've been very lucky. I know you're all going to say it was my fault for not making a backup of the text file - but that required not only a special lead, but special software - it's a long story - and I had neither at the time. (I mistakenly thought the file would be safe inside the computer.)

I have found a biro and a piece of paper very handy after that - they're a great alternative to my usual word processor. But I find I make too many errors in my handwriting - it's either unreadable after so long, or I need to cross whole words out all over the place - even whole sentences, which would have been erased easily if I'd have been writing my text on a computer screen instead. It's great to be able to type away and use the Delete key for any errors you might have made - that way, you can be sure the end result will be free from mistukes! (That was deliberate, please don't write in to tell me.)

Moving on, I'd better be doing just that. Our trustworthy Editor says he wants another page like this one for his next issue, so I'd better come up with some ideas, er, I mean get it ready... I'll see you around then everyone - take care. Your servant, The Shredder.

NEWS

VIRGIN FORWARD

A special report on one of Britain's most famous success stories, Virgin - Bring out the Branson! By Christopher Hester

Richard Branson's Virgin empire continues to grow. In three different items of news here, Virgin are emerging still as a youthful, expanding company whose products have grown from the record label of the Seventies, to cover everything from airlines to cola soft drink.

Into The Future

Branson's baby has been busy gaining yet more coolness points than before, with the launch of Virgin's latest project, The Raft. Using the electronic network of computers known as the Internet, Virgin have set up their very own site for information. In this case, it's specifically to do with Virgin Records. (Remember *Tubular Bells?*) The Raft is designed to take advantage of the part of the Internet known as the World Wide Web, where graphics and even sound samples can be loaded into your computer, via a modem and the telephone line. But The Raft is alleged to be more than just a quick method of gaining info on the latest Virgin releases, with assistance from other labels as well, together with the Royal College Of Art, and other names and artists. The result is hoped to be a futuristic example of an exciting place to hook up to with your computer. The Internet address you need to dial into is:

http://www.vmg.co.uk

An explosive venture

A joint venture between Virgin and Norwich Union has resulted in the Virgin Direct Personal Financial Service. Details remain sketchy at the moment. However, at a celebration party held recently, Richard Branson turned up in person. Whilst opening a bottle of champagne, he reportedly sprayed three brand new computers, blowing up the monitors!

David And Goliath?

The biggest new scheme to arrive from Virgin however, has to be the release of Virgin Cola. This is, of course, a competitor to every other soft drink cola on the market. While Sainsbury's might be said to be happy with the sales so far of their own-brand cola, that caused a lot of fuss when it was launched a while ago, Richard Branson has set his sights right at the top; he wants Virgin Cola to eventually take over from

Coca-Cola and Pepsi, as the No.1 seller, at least outside of America. But it'll be a big fight. Pepsi and Coke together have around 78% of the market, with own-label brands of cola claiming a healthy 14%. Other brands make up a further 4%, as does Virgin's cola so far * - not bad, but hardly enough to beat the top makes. In fairness, it's a new brand that has yet to realise its full sales potential. Branson has done well with it so far, and seems agressive in his tactics, finding niches for the drink where it'll make the most impact. If he's to achieve his aim - to topple giants Pepsi and Coke - he'll need not only strength and power, but a lot of luck.

(* Source: Taylor Nelson AGB Impulse, as quoted in SuperMarketing 12th May 1995)

Pretty Good Year
The Editor chooses his favourites from last year...

Best Albums Of 1994

After much contemplation and effort, I decided it was impossible to choose between certain albums, in order to create a Top Ten list of favourites - there were so many! A year in which nearly all my favourite acts seemed to release an album! Yet one album shone above all others in terms of sheer excitement, and that rock 'n' roll spirit. That album is...

1. Oasis - *Definitely Maybe.*

The rest of my choices are as follows, in no order... Yes - *Talk*, Marillion - *Brave*, Ice Cube - *Bootlegs & B-Sides*, Ian McNabb - *Head Like A Rock*, Sugar - *File Under: Easy Listening*, Pink Floyd - *The Division Bell*, Neil Young - *Sleeps With Angels*, Kristin Hersh - *Hips & Makers*, Beck - *Mellow Gold*, The Stone Roses - *Second Coming*, Nine Inch Nails - *The Downward Spiral*, Elvis Costello - *Brutal Youth*, R.E.M. - *Monster...*

Also recommended... Consolidated - *Business Of Punishment*, Prince - *Come*, Public Enemy - *Muse Sick-N-Hour Mess Age*, Manic Street Preachers - *The Holy Bible*, Jon Anderson - *Change We Must*, Lightning Seeds - *Jollification*, Tori Amos - *Under The Pink*, Terrorvision - *How To Make Friends And Influence People*, Throw That Beat! - *Superstar*, Credit To The Nation - *Take Dis*, David Byrne - *David Byrne*, Portishead - *Dummy*, The Black Dog - *Spanners...*

Best Soundtrack Album Of 1994

No contest at all, really... Various Artists - *Natural Born Killers*

Basic Machine Code Techniques

by Robin Rawson-Tetley

OK, what is the hardest part to learn while tackling the intricacies of assembly language? The instruction set? The structure? Yes, you, at the back, correct! It's getting a technique that is the hardest part. Obviously I can't demonstrate every way of doing things, or give you a technique as it takes a long time to develop your own - one of the reasons why assembly is so personal compared to other languages, ie: everybody does it differently. What I can do is show a few routines that any assembly programmer will definitely require and how to do certain things. Note that the thing here being demonstrated is the technique, you must know the instruction set first! Also, if you work in a different code and know 6502, it shouldn't be too hard to translate.

The first routine is fairly simple, just a loop, it has an 8 bit counter and can therefore only count from 0 to 255.

```
        ldx #200      ; count value.
        stx count
loop    .. what you want to loop
        dec count
        bne loop
        .. continue
count   .byte 0
```

This is if you just want to do something more than 255 times (a nested loop). You can determine the value by outer x inner.

```
        ldx #50       ; outer
        stx outer
resetin ldy #60       ; inner   50 x 60 = 3000 times
loop    .. what to loop
        dec inner
        bne loop
        dec outer
        bne resetin
        .. continue
inner   .byte 0
outer   .byte 0
```

How the routines work:

The counter value is stored in memory, your loop program is executed, then the counter value is decremented with the DEC command (this is so you can use all the registers in the loop program as DEC works on memory only). If the DEC makes the counter zero, the zero flag is set - hence the BNE instruction, only send back if the result is not zero. Pretty simple? Loops are extremely common in a program and a short routine like this one can save valuable memory. In the second program, you can see that once the inner loop has finished, the outer is decremented, if it is still not zero, the inner loop starts again. Programming, however is rarely that simple, usually you want to count memory locations, or use your loop to do more than repeat one bunch of instructions over again. Imagine you wanted to check memory locations 49152 to 52482 for a byte $ff (255)? You could combine the above nested loop with a simple memory count, eg:

```
          lda #<49152
          sta $fb
          lda #>49152
          sta $fc         ; 52482-49152=3330
          ldx #30         ; outer ; 3330/30=111, therefore inner=111
          stx outer       ; outer = 30
resetin   ldy #111        ; inner
          sty inner
          ldy #0          ; reset y
loop      lda ($fb),y
          cmp #$ff
          beq matchfound
          inc $fb
          bne endlp
          inc $fc
endlp     dec inner
          bne loop
          dec outer
          bne resetin
          ; match not found if here.. (loop expired)
matchfound ..address found at is in $fb/c
inner     .byte 0
outer     .byte 0
```

How did that work!? First of all, we set the start address of 49152 into $fb and $fc - if you are not familiar with this, this is called splitting a sixteen bit

number into two parts - the Most Significant (high byte) and Least Significant (low byte) (both are below 256), you stored the hi byte in $fc (your number rounded up to the nearest number of 256's) and the lo byte in $fb (the remainder of your number / 256). Imagine dividing your number (which was 49152 or $c000) by 256 on paper - you would find out how many 256's went into it, and then have a remainder - the number of 256's divisible is the high byte and the remainder is the low byte - if you try it for 49152, the low byte is zero (256 goes in exactly, which you can see from the hexadecimal equivalent), and the high byte is $c0 or 192 because 192 x 256 = 49152. We put this in the zero page so we could do something called "post indexed zero page addressing" (!), what this means is when the computer got to the instruction "LDA ($FB),Y", it looked at the address held in $fb (note that the lo byte always goes first), then added the value of Y to it to get a new address (or not in our case as Y was deliberately zero all the way through), and put the contents of that address in A for us to compare. The rest of the routine is fairly simple. (There is another type similar, but it uses the X register and is in the form "LDA ($FB,X)" and this is called pre-indexed zero page addressing, as the X register is added onto the zero page address before the memory location held there is obtained.)

That routine could have been a lot shorter - we could have checked for the address reaching the end instead of a nested loop, or (watch how I switch subjects like oiled gears!) we could have used Self Modified Code. What is self modified code? Basically your program edits small parts of itself to save memory and processor time - this is the above routine again but with self modified code and no nested loop.

```
        lda #<49152
        sta pointer+1
        lda #>49152
        sta pointer+2
pointer lda $abcd        ; this gets the data and is self modified
        cmp #$ff
        beq matchfound; goto matchfound if $ff found
        inc pointer+1
        beq passo        ; go here if passed 256
loop    lda pointer+1
        cmp #<52482      ; lo byte fin?
        bne pointer
        lda pointer+2
```

```
        cmp #>52482    ; hi byte fin?
        bne pointer    ; if not go back
        ; match not found if here.... (loop expired)
matchfound ..continue
        bne loop
```

See how short that was? So how did it work? When the code is assembled, the pointer label points to the instruction LDA 49152, but the assembled code is assembled into the format of instruction code, then the aforementioned Lo and Hi byte denominations, so pointer+1 is the lo byte and pointer+2 is the hi byte! All we are doing is incrementing these and saving buckets of processor time, and not using the zero page AND not using any other external memory locations as we are using the address to count (notice the end - if the lo byte is the same as the finish lo byte, we check the hi byte - if that is the same as the finish hi byte we finish). Notice that using the above code has saved both processor time and memory.

Where else might you use self modified code? Well I'm sure there are thousands of uses for it that I haven't thought of, but a great place to use it is in raster interrupts - they have to be quickly executed and you always have to save the registers before a raster interrupt routine. You can save time by doing this at the start:

```
        ; example raster interrupt routine
        ; set split etc. here...
raster  stx xs+1
        sty ys+1
        sta as+1
        inc $d019
        ..rest of routine etc.
xs      ldx #0
ys      ldy #0
as      lda #0          ; ALL SELF MODIFIED ABOVE
        rti / jmp $ea31 ; (no BASIC+KERNAL / BASIC+KERNAL)
```

The Jump Table:
Sometimes, in a program it is necessary to jump to one of a number of routines, based upon a value. This is where a jump table comes in handy, as an example, you enter the following program with A holding the number of the routine you wish to jump to (note that routine 1 is number 0):

```
jmptab asl a          ; double it first, two bytes required for
                      ; an address
        tax           ; make it the index
        lda tab,x     ; get the first byte
        inx           ; increment the index
        lda tab,x     ; get the hi byte
        sta hib       ; and store it
        jmp (lob)     ; JMP to the address
lob     .byte $00     ; this is the jump vector
hib     .byte $00
tab     .word $c482   ; this is the address table
        .word $8436   ; etc.
```

Note that JMP is the only instruction which has the indirect (in brackets) address mode allowed. It means JMP to the address held at (address). As an example, if the routine was entered with 1, it would jump to routine 2 at $8436 (as 0 counts as the first routine).

Using the equivalent of < and > in 6510.
Sometimes in a program, you may want to test if a number is greater than or lesser than another number, there are two ways to do this. First of all you put the first number in the accumulator, then use the CMP instruction with the second number.

Method 1:
If the second number is greater or equal to the first number, the sign flag will be set to "plus", otherwise it is set to "minus".

Method 2:
If the second number is greater than the first number, the carry flag will be set as the machine is saying it will require a borrow if you were subtracting the two numbers, otherwise the carry flag will be cleared.
Example:

```
        lda #10
        cmp #10
```

The zero flag will be set because they match, the sign flag will say plus because it is equal/greater than. The carry flag will be cleared.

lda #10

cmp #12

The zero flag will be cleared, the sign flag will say plus and the carry flag will be set.

lda #10

cmp #4

The zero flag will be cleared, the sign flag will say minus and the carry flag will be cleared. This method also works with the index registers although the carry will be set if the number is equal or above the original number if you use the index registers.

Best Singles Of 1994

Again, far too many to produce any kind of sensible list. Except for...

1. Tori Amos - *Cornflake Girl*
2. Manic Street Preachers - *Faster*

After those, the rest are as follows - again, in no order... Crash Test Dummies - *Mmm Mmm Mmm Mmm*, Marxman - *The Cynic EP*, Tori Amos - *Past The Mission*, Terrorvision - *Middleman*, Massive Attack - *Sly*, Bruce Springsteen - *Streets Of Philadelphia*, Crowded House - *Fingers Of Love* (and *Pineapple Head*, if that was a single last year - was *Locked Out* the year before, though?), The Stone Roses - *Love Spreads*, Oasis - *Whatever*, Credit To The Nation - *Teenage Sensation*, Marillion - *The Hollow Man*, The BC52s - *Meet The Flintstones*, Public Enemy - *Give It Up*, R.E.M. - *What's The Frequency, Kenneth?*, Stiltskin - *Inside*, The Cranberries - *Linger*, Soundgarden - *Black Hole Sun*, Warren G - *Regulate*, Tori Amos - *Pretty Good Year*, Oasis - *Live Forever*, Terrorvision - *Oblivion...*

Worst Single Of 1994

Anything by Boys II Men!! Or that godawful Rod Stewart\Sting\Bryan Adams thing!!

Best Cover Version Of 1994

Wet Wet Wet - *Love Is All Around* / Gun - *Word Up*. (You decide!)

Special Award For Outstanding Production 1994

Trevor Rabin, for his Production of Yes - *Talk*.

Special Award For Outstanding Use Of Technology 1994

Trent Reznor for Nine Inch Nails - *The Downward Spiral*.

How To Make A Thingy (Part VII)
by Stephen Groves

TESTING, TESTING.

So, what do we need to get this working, well we need to set up a few
more variables at the beginning of the program. The following few
lines of code should be inserted right at the very beginning of your
code, just after the ORG 28000 statement.

```
MAINOB      EQU 601DH          ;The position of the main
                                object table
FLAGS       EQU 63C4H          ;The position of FLAGS

RAMBUF      EQU 6B67H          ;The position of the RAM
                                buffer
LOCTB1      EQU 658BH          ;The position of the
                               ;location tables
LOCTB2      EQU 6621H          ;

            DI                 ;Disable the interrupt

            RES 5,(IY+01)      ;Reset the ROM bit that shows
                                a key has been pressed
            LD HL,(5C3DH)      ;The address ROM uses for
                                errors
            PUSH HL            ;Store it on the stack

            LD HL,ERROR        ;Substitute our own

            PUSH HL            ;Put it on the stack

            LD (5C3DH),SP      ;Put it where ROM can use it

            LD IX,FLAGS+128    ;Point IX at the middle of
                                FLAGS. IX uses the offset
                                of -128 and +127
            LD A,1

            LD (IX+0AH),A      ;Load the location flag with
                                the first location
            XOR A              ;Put zero into A

            OUT (0FEH),A       ;Make the border black

            LD A,6

            LD (IX+11H),A      ;Set the maximum number of
                                items carryable
            LD (5C8DH),A       ;Permanent screen colour

            LD (5C48H),A       ;Colour for bottom 2 lines

            LD (6C8FH),A       ;Temporary colours

            LD HL,UDG          ;Our UDG for the border

            LD (5C7BH),HL      ;The UDG variable in Basic
```

```
            LD A,2              ;By sending 2 into this ROM
                                ;routine, we state that
            CALL 1601H          ;printing in ROM should
                                ;appear on the main part of
                                ;the screen
            JP MAINRT           ;The routine that will
                                ;control the order of all
                                ;the other routines
ERROR       POP HL              ;Removes value from stack

            LD (5C3DH),HL       ;Restores it to BASIC
                                 variables
            LD HL,2758H         ;Required in 'H'L before
                                 returning to BASIC
            EXX                 ;Exchange registers

            EI                  ;Enable interrupts

            RETI                ;Return to BASIC
```

Notes on ERROR

ERROR is a routine that we will get to much later in the program. It
deals with errors in loading and saving to tape. I have cheated
slightly by making the routine just return to BASIC at the moment, but
then we aren't using the cassette just yet.

OK then, back to testing. You should have noticed in the code above
that we make a jump to MAINRT. You may even have noticed that we
haven't got a MAINRT yet. In fact we're going to have two pieces of
code called MAINRT. The first will purely be for testing what we have
written so far, the second will be part of the main program and will
replace the first one.

```
MAINRT      CALL CLS            ;Clears the screen

            CALL SETUP          ;Prints the border for the
                                 graphics
            LD B,0              ;Zero into B

            LD C,(IX+0AH)       ;BC now contain the first
                                 location number
            LD HL,LOCTB1        ;HL points at the start of
                                 LOCTB1
            ADD HL,BC           ;(HL) now points at the first
                                 part of the address of the
                                 text for the location in BC
            LD E,(HL)           ;Put it into E

            LD BC,96H           ;96H is the length of LOCTB1

            ADD HL,BC           ;(HL) now points at the second
                                 part of the address for the
                                 location text
            LD D,(HL)           ;Load it into D

            EX HL,DE            ;Move it into HL

            JP PLOCR            ;To print the location text
                                 and return via ERROR
```

Right then, now we're getting closer to the moment of truth for this
first part of code. Assemble what we have done so far, and save the
resulting machine code. Hopefully you shouldn't get any error
messages.

To test what we have assembled, we obviously need a small piece of
text, some exit arrows, and some objects to show SMINV. We are going
to provide all this in a small BASIC program.

```
10 REM First we put in some text.
20 FOR X = 30000 TO 30077
30 READ N : POKE X,N
40 NEXT N
50 REM Then we put in two objects.
60 FOR X = 40000 TO 40017
70 READ N : POKE X,N
80 NEXT N
90 REM Then we put some entries in the tables.
100 POKE 25391,107 : REM for some EXIT arrows.
110 POKE 25996,48 : POKE 26146,117 : REM for LOCTB1 & LOCTB2
120 POKE 25291,255 : POKE 25292,255 : REM (255) shows that the
    items are carried
130 POKE 25796,64 : POKE 25896,156 : REM OBJTAB + 1 & + 101
140 POKE 25797,73 : POKE 25897,156 : REM OBJTAB + 2 & + 102
160 DATA 84, 104, 105, 115, 32, 105, 115, 32
170 DATA 106, 117, 115, 116, 32, 116, 111, 32
180 DATA 116, 101, 115, 116, 32, 105, 102, 32
190 DATA 116, 104, 101, 32, 112, 114, 105, 110
200 DATA 116, 105, 110, 103, 32, 114, 111, 117
210 DATA 116, 105, 110, 101, 32, 119, 111, 114
220 DATA 107, 115, 32, 105, 110, 32, 34, 72
230 DATA 111, 119, 32, 116, 111, 32, 109, 97
240 DATA 107, 101, 32, 97, 32, 84, 104, 105
250 DATA 110, 103, 121, 46, 34, 13, 115, 119
260 DATA 111, 114, 100, 32, 32, 32, 13, 114
270 DATA 105, 110, 103, 32, 32, 32, 32, 13
280 RANDOMIZE USR 28000 : REM To test the code.
290 PAUSE 0 : REM To preserve the screen.
```

If you load the code and then BASIC program into memory and run it,
you should get some idea of what the finished adventure screen will
look like.

Letters

Dear Chris,

What an excellent postal delivery I had today. One letter from Spectrum enthusiast Jon Rose (ex-editor of ETM tape magazine), ADVENTURE PROBE and ADVENTURE CODER. This new issue of CODER is possibly the best so far. Nicely presented and loads of articles for me to read at leisure. Your writing style is better than ever, and I don't mind the interval between issues if the result is as good as this. I hope you get a decent mention about it in the next issue of PROBE, as you deserve a bit more recognition.

Right, now for my own comments of the 8-bit 'life and death' situation: Your article seemed quite fairly balanced. It really boils down to what the user needs and is content with. I'm still a loyal, and fanatical, SAM owner. I still have my old, pre-Amstrad 128k, which looks like the Spectrum + but with the extra memory and heat sink on the right hand side to keep my hand warm on cold days. It has a Plus D system attached to its wobbly old interface as well as the optional cassette player, all mounted on a board as one unit. The trusty old Spectrum still works very well, but is mainly used for tinkering about with PAW databases so they can be converted to SAM.

My SAM set-up has become quite impressive, even if I say so myself. The actual SAM is 512k with an additional one megabyte memory module, colour monitor with stereo sound, twin disk drives, COMMS interface, joystick, attached cassette (not really needed) and a mouse. My SAM uses two printers: a Star LC 10 and a Canon BJ 200. For good measure, and extra insurance in case of a computer fault, I've recently bought a brand new 512k SAM with a single drive. These SAM drives are Citizen 3.5" slot-in type, therefore are very easy to swap around, as are the peripherals I use with the system.

SAM uses 16 colours selectable from a range of 128, which is ample for my needs. Line interupts can be used to show more of these colours on screen at any time. Using SAM's MasterDOS and MasterBASIC programs SAM can be quite similar to a PC in file handling. Subdirectory structure is similar to the PC, a clock on my SAM's motherboard allows time and date functions and the BASIC is one of the best of any 'normal' home computers, with auto line numbers, word-skip, search and replace, line splitting and joining, powerful PROCEDURE capabilties, and loads more. The amount of SAM software is now quite impressive, even though most is written by enthusiasts for the enjoyment of it. We've loads of good value word processors, assemblers, databases, monitor

programs as well as more and more arcade games. Even adventures on SAM are quite good, with another two on the way, as far as I know. New hardware which I certainly know is being developed include a hard drive and a MULTIROM, which will slot into SAM's motherboard with 'built-in' utility software and snapshot features. (See? I said I was a SAM fanatic!)

I'm a packaging technologist by trade (i.e. I design cardboard boxes!) and so I have almost exclusive use of a 486 PC with math-coprocessor, VERY good hi-res monitor, colour scanner, modem and a industrial-size plotter. This plotter could allow me to do adventure maps up to 900mm by 1200mm in size, if the boss didn't keep walking past. I mainly use some CAD software called FASTCAD for my job which is lovely to use for all sorts of technical drawing. (I design box styles and shapes rather than artwork.)

The difference between the SAM and PC systems is enormous when comparing memory, speed and graphic capabilities, but both are well-suited to my purposes. For home use I need a good keyboard and monitor, word processor, fair graphics power, disk storage and ease of use, as well as a rugged and reliable system as I don't want hefty repair bills every few months. My SAM fits this bill to perfection. The office PC operates to the level I need, with fast speed, very finely detailed screen image and loads of pretty colours to play with. However, the equipment is very expensive, with even the plotter costing £4,000 (and it's at the cheaper end of the range.) Despite this price, the office PC has been very prone to problems, such as the monitor's red gun failing, two faulty mice so far, one duff keyboard, three duff data switches, heat-creeping of the chips needing regular attention, a failing cache memory and a rather dodgy modem. We've only had it for 18 months, so it isn't exactly ancient. Maintenance bills have been well over £1,000 so far, so I'm not in any great hurry to buy one for home use. Touch wood, my SAM set-up has worked without a single fault for nearly four years.

We gained a second office PC a few months back, to run alongside my one. After a fortnight it decided not to believe it had a floppy disk drive, ignore the mouse and lock-up ever half hour or so. The repairman said that he's have to take it back to the manufacturers for a day or so, but that all the files would be preserved. Later the same day, he rang in to say that the computer self-destructed, wiping the hard disk and obliterating three boards of some sort within the unit. To my amazement, the makers simply junked the whole thing and sent a replacement. I asked why they hadn't kept the original to find out the fault and they said it wasn't worth it, as a new model was coming out in a few months time. I pity any poor buggers who may have such a machine at home, which may self-destruct after the warranty is up. One of the main things I dislike about PCs is the rate at which they're upgraded, so making older ones almost obsolete as software is incompatible with them.

At least my SAM can read PC text files, and vice versa, so I can easily type letters at work and print them out/use them on SAM once I get home.

Snap! I also have an Amstrad Notepad (NC 100). They're lovely, friendly machines to use. Again, this is a computer that suits me, as I really just want a handy and portable word-processor. I don't need some laptop costing over £400 with WINDOWS running on it. I'm currently working on a method of transferring text files between the Notepad and SAM, as both are well-suited to transfers.

If my SAMs ever exploded, and couldn't be replaced, I'd probably go for a PC, but only out of necessity. They're widely catered for, and should have good staying power, if they're well made. There are times that I think PCs and their software are more than I need. When writing, I rarely do more than twenty pages at the most, so small 8-bit word processors are all that I need, not huge, expensive programs on a PC. Likewise with databases and other utilities.

When it comes to adventures, I tend to prefer text-only, so just a few colours are OK. Anyway, SAM's 512k memory is enough for most decent games, I reckon. Some of the graphics in games on bigger machines, within adventures, may be impressive, but I start doing my sums thinking that for every 48k used on them I could have a Jack Lockerby adventure, or similar.

Of course, these are my own feelings, and I'm sure other readers, and yourself, may really appreciate the powers of the bigger machines, and use them a great deal. That's fine. It'd be a shame if we all wanted the same boring things. As long as we find computers that we're happy with, that is an ideal state of affairs.

Onto more adventure-based stuff for a while: I think I may have a few snippets of news/information worthy of inclusion in CODER some time.

Martyn Groen, a SAM enthusiast in Holland, is busy converting Spectrum PAW to SAM disk compatibility. I believe that Gilsoft have no objections to this, as long as only existing PAW users are allowed to own it. (Hopefully they may be tempted to relax this condition.) This conversion isn't just a snapshot-type conversion, but is a fully working version of PAW. SAM can rarely run Spectrum 128k programs unless major re-coding is done, as the old Spectrum 128k memory layout is different from SAM's. Martyn has achieved this with PAW so that both 48k and 128k versions of the utility run on SAM. Also, some simple routines allow Spectrum taped PAW games to be converted by SAM into disk files, then re-loaded as a disk-based game, saving and loading like the Plus D system.

I've seen several versions of this PAW conversion, and I know that it works extremely well. I've converted many 128k PAW games to SAM and it's great news for adventurers who own SAMs. Enhancements are being made to PAW to allow games to incorporate SAM music and maybe SAM Mode 4 screens, using EXTERNS. Martyn is even looking at ways to re-configure the PAW utility to access

all of SAM's 512k memory. Good news indeed! I'll try and get
more detailed information to you when it's released, as
supplementary technical notes already exist within the disk all
the PAW utilities are on.

While dabbling with the SAM ADVENTURE SYSTEM, I discovered
that the location database, and others, can be loaded into
several of SAM's word processor packages, which can allow easier
editing, spell-checking, etc. Anything which makes life easier
is OK by me. Another SAM game written using the SAS, called
SMUGGLER'S HAUNT, is now in its final stages, and is at the top
of my list of priority buys.

For those SAM adventure writers wanting graphics within a
game, the SAMPAINT art package is perfect, as it has many
features normally seen in 16-bit or 32-bit machines. Anti-
aliasing, plasmas, block-blending, perspectives, gradients,
tracing and other functions exist, but it'll take any user quite
a while to learn the dozens of features within it, as it is way
ahead of packages such as ARTIST II on the Spectrum. SAM can
also have AMIGA, ST and some PC graphics ported over on to it
using some of the many conversion programs now available.

SAM is still very much a minority machine, but it's still
selling fairly well, and has built up a good, loyal following.
It is probably classed as a cult computer, but it's alive and
doing modestly well, and its survival is in the hands of its
enthusiastic users, rather than commercial software companies
and magazines. I think this is why interest in SAM has continued
to grow while the poor Spectrum declined swiftly as both
software houses and magazines withdrew support. I've even
recently been sent some copies of a SAM disk magazine in the
Czech Republic, and SAM seems very popular over there, judging
by the games and programs on the disk. What a shame all the text
is Czech! I wonder if the Czechs play adventure games....

I see I've typed nearly four pages so far, which may already
be boring you, so I'll draw to a close! Thanks again for such a
superb edition of CODER, and good luck with the next.

All the very best,

Phil

Phil Glover.

Dear Chris Hester !

Although there are several thousand kilometers between us,
I hope for your help and responsiveness !
I am a great admirer of ZX Spectrum, especially of adventure
games on it. Perhaps I am slightly touched, but I would like
to receive from you your magazine "Spectrum Adventure Coder"
or "SPAC" (for acquaintance) or any information about Adventu-
re Games (catalogues, booklets, clues books etc). Also very,
very much I would like to see with my own eyes some magazi-
nes about ZX Spectrum, namely: Your Sinclair, Sinclair User,
ZX Computing, Your Spectrum, Crash, Your Computer or Section
"Adventure Games"/"The Sorceress" from them, and what's more
in any form: copies, microcopies or microfiches.
    Besides I am crazy about Michael Moorcock and his fantasy
works and at the present time I look for any his series or
books and I am fully confident that some his works were con-
verted into adventure games (but what names namely?).
    Also for long time I look for "The Secret Diary of Adriane
Mole" by Sue Townsend, "The Fourth Protocol" by Frederick
Forsait, "The Unorthodox Engineers: The pen and the dark" by
Colin Kapp, "My Secret File" by John Astrop (last both from
Mosaic Publishing Ltd), "Dennis Through The Drinking Glass"
(with sequal) and "Archimedes' Magic Screw" from Applications
software specilities (as adventure games, as their originals
in books)...and at last real serious work on Adventure Games
from U.K. and other countries (perhaps,one of Atlas Adventure
Software?).

                I shall be most grateful
                for your help and with
                many thanks to you and
                Yours Sincerely
                        German Sthetnikov ->

29 December, 1994

Dear Chris,

Thank You for Adventure Coder #22, it's nice to see the mag. is still going strong. I hope all's well with yourself and here's wishing you a very good New Year.

As you may have guessed I've got myself a new wordprocessor, Wordworth3.1 which together with the bubblejet can produce some decent looking documents. Wordworth is very good, but still doesn't match the power of something like Wordperfect or Word for Windows.

To be honest, I'm starting to get worried about Commodore's prolonged buy-out/liquidation or whatever. Christmas has been and gone without Amigas in the shops. Mail order companies are concentrating on 3rd party peripherals and are beginning to advertise PCs, which makes sound business sense. What worries me is being left out cold if Commodore go under and take Amiga with them. The peripheral market would keep the existing users going for a while, but eventually it will decline as the Amigas become outdated. Users would have 3 choices:

1. Keep expanding their system with RAM, CDROM, Hard-drives, accelerators and stick with what they have. (Fine if they don't play games.)

2. Keep the Amiga but don't expand, just save up for the PC.

3. Sell the Amiga ASAP and buy a PC.

A decent 486 PC costs around £1000, an outlay I couldn't really afford or justify, since my own set-up is just what I need. Except maybe a decent colour printer...

Hopefully Commodore will sort things out, but they have to be quick or else the market will have sunk beyond hope. Whatever happens, I'll write an article for Coder. It's not only 8-bitters who have problems!

I've taken a break from Adventuring over the last few months, concentrating on getting a lot of DTP done for my church, Christmas being a busy period and all that. I've also had a few stories printed in Mark Rose's Alternaties magazine which is getting itself quite a good reputation (despite my stories!).

I ought really to get back into the Adventure scene. I ceased subscribing to Adventure Probe when Mandy Rodrigues ceased to be editor. I don't know if Probe is still going or what else is happening in the scene...

I agree with your comments regarding printers, I would never be without my trusty BJ10sx - it's light, quiet and overall has excellent print quality, although I would never call it laser quality (unless you use some special expensive paper) like they used to in the adverts. Near-laser is more correct. Talking about lasers, Panasonic do a laser at about £299, but there is a word of caution. Have a good look at the Tech.-specs for these cheap lasers because they might not be lasers at all. If you see the term "LED" then you don't have a true laser. Instead of a laser light source, you have a cheaper LED (light emitting diode) source. Whilst producing good results, they may not be as good as a true laser, although for you and I, the difference isn't worth worrying about. Worry when you get to the 600dpi resolutions.

As for your comments in "8-bits-dead or alive", I agree with most of them, especially the Notebook which was featured in Amiga magazines because of Protext and portability. When I get the chance, I'll have to write something in reply, to keep the open discussion going, although you seem to have been pretty comprehensive.

Very Best wishes,

John Ferris
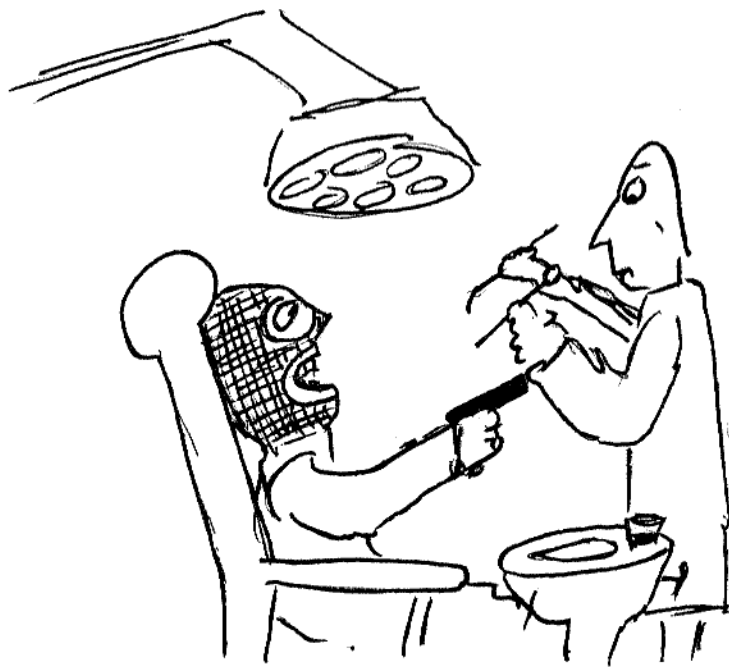
"OK! This is a check-up!"

BACK ISSUES

These are available at the following reduced prices:

£1.00 - Britain   **£1.40** - Europe
£1.85 - Zone 1 (Canada, Egypt, Falklands, Saudi Arabia, USA, etc.)
£1.95 - Zone 2 (New Zealand, Australia, etc.)

Payment is by the usual manner - cheques payable to "C.Hester", at the usual address, or you can use a Postal Order, International Giro, or British decimal stamps for small amounts. If you want more than one back issue, then just mutliply the cost by the number of issues you want - for example, 5 issues (if you live in Egypt) is 5 times £1.85, equalling £9.25.

ISSUE 1 - Utilities & Add-Ons, GAC+ review, PAW (available exits), Z80 Machine Code (Z80 overview), Whatever Happened To... *Valley Of The Source*, GAC graphics (colour, perspective, ellipses, rectangles), Useful Addresses, etc.

ISSUE 2 - GAC (pokes, starting a game), GAC+ (pokes), PAW (parser, vocabulary), Z80 Machine Code (command input routine), Whatever Happened To... sound-only games, STAC (starting), Adventures (storyline, writing), Useful Addresses, Utilities & Add-Ons, etc.

ISSUE 3 - The A-Z Of RPG, PAW (cars, again, oops, find object, overlays), GAC (doors), STAC (tips, list), Spectrum Machine Code (parser), Useful Addresses, Utilities Available, etc.

ISSUE 4 - PAW (inventory, get all, drop all, containers, objects in mazes, taxis, exit printing, swear protection, again, oops, clocks, add-ons), Adventure Columns reviewed, GAC (inside info), Useful Addresses, Utilities Available, etc.

ISSUE 5 - PAW (characters, objects, telephones, examine, password, role-playing, clocks), ADLAN (objects), Play-By-Mail, Atmosphere, Tom Frost interview, Lastability, etc.

ISSUE 6 - PAW (flag addresses, mazes, password, inventory + objects, characters, start anywhere, lives), GAC (messages, character sets, wear, exit printing, characters, save + load, start-up, bugs), Whatever Happened To... Isaac Asimov adventures, ADLAN (messages), Adventures (endangered?), etc.

ISSUE 7 - BUMPER ISSUE!! - PAW (flag directory, money, oops, him/her, last), ADLAN (vocabulary), 6502 Machine Code (registers, hex, useful memory addresses), Sexism, Sandra Vogel interview, Patrick Walsh interview, Adventures Of The Future, Lastability, etc.

ISSUE 8 - PAW (exits, mazes, characters, overlays, fonts, flags, object weights), Adventure Writing For Beginners, ADLAN (light + dark), Spectrum Adventure Utilities, GAC (character with an inventory), *Adventures On The Spectrum* (Mike Gerrard) review, 6502 Machine Code (modes), New Words, STOS (ideas), etc.

ISSUE 9 - PAW (independent characters, chance/random, exchanging objects, externs, passwords, memory map, overlays), GAC/PAW (converting commands), Z80 Machine Code (room descriptions), STAC (thoughts), 6502 Machine Code (graphics screen, stack, useful commands), CBM64 Art Programs (+ useful routines for them), ADLAN (errors in manual), etc.

ISSUE 10 - PAW (character speech, do's + dont's, externs), Z80 Machine Code (exit printing), New Words, 6502 Machine Code (interrupts, handy BASIC + machine code routines), 1541 Disk Drive (secrets, sorting files, Toolkit IV secrets), ADLAN (6-room game!), STOS (adventure creator project + routines), etc.

ISSUE 11 - STAC (starting a game), PAW (input editor, characters, 128K into 48K conversion), The Ultimate Adventure Creator, *Gems* compilation review, ADVSYS review, 6502 Machine Code (routines including sprites in the border!), ADLAN (explanation of last issue's game), TALESPIN (thoughts), Adventure Languages, etc.

ISSUE 12 - STAC (plots, colours), Customization (hardware + software!), PAW (characters, screen printing, flags, externs (sfx)), 6502 Machine Code (loaders, protection), ADLAN (give-to), Z80/68000 Machine Code (converting commands), New Words, etc.

ISSUE 13 - PAW (inventory with objects inside containers listed, characters, objects + containers), Programming Versus Writing, Z80 Machine Code (locations, objects), ADLAN (character sets), Virtual Reality, *Computer Adventures - The Secret Art* review, etc.

ISSUE 14 - CES Show review, Advice For Beginners, PAW (telephone system), ADLAN (starting), Humour In Games, *Splatt!* review, *A Beginners Guide To Adventures* review, Z80 Machine Code (objects, inventory + more), etc.

ISSUE 15 - Mike Gerrard interview, PAW (characters), Adventures (experience, writing styles), ADLAN (variables), etc.

ISSUE 16 - currently out of print! Sorry!

ISSUE 17 - currently out of print! Sorry!

ISSUE 18 - ADLAN (pictures), Tips (Easyscript on the CBM64), Thingy part 2, Creating Puzzles, etc.

ISSUE 19 - currently out of print! Sorry!

ISSUE 20 - STAC (animation, on-screen exits), The Taxman Chronicles part 2, BASIC (strings), Puzzles, *Adventure Link 2* review, PD, Thingy part 4, AMOS (writing a game), PAW (Technical Guide), etc.

ISSUE 21 - GAC (punctuation, descriptions, commands, containers, messages), STAC (wear/remove, objects, combat, colour, WITH command), Polish Computer Magazines review, Thingy part 5, BASIC Text Compression, etc.

ISSUE 22 - Groovy 6510 Adventure Routines For The C64, Contrived Plots, STAC (examine, search, containers, rooms) The Taxman Chronicles part 3, 8-Bits - Dead Or Alive?, AMOS, Thingy part 6, etc.

Regarding the issues out of print at the moment, I am hoping to get these reprinted as soon as I can.

There are also back issues available for the four issues of *Adventure Workshop* that were produced as a sister magazine to this one. They were primarily concerned with 16-bit computers, here's what was in them. (The price is exactly the same as the back issues of *Coder* above!)

A.W. ISSUE 1 - TALESPIN (thoughts), STAC (starting), Programming Versus Writing, AGT review, Z80/68000 Machine Code (converting commands), STAC (thoughts), CES Show review, AMOS (parser, etc), *The Blag* review, *Computer Adventures - The Secret Art* review, etc.

A.W. ISSUE 2 - TALESPIN (starting), STAC (starting, User Routine Access Protocol by Sean Ellis, author of STAC!), AMOS (parser, graphics), Writing Styles, ADVSYS review, Humour In Games, Virtual Reality, Z80/68000 Machine Code (converting commands cont.), etc.

A.W. ISSUE 3 - Magnetic Scrolls exclusive interview!!!, STAC (writing a game, list of faults, exit printing), AMOS (data + editors), TALESPIN (running a game), ST AGT review, Adventure Standards, 68000 Machine Code (addressing modes with table), HATRACK II review, etc.

A.W. ISSUE 4 - Creating Puzzles, Mike Gerrard interview, STAC (object printing), Tips (Star LC-10 printer), Writing Tips, HATRACK II details, GAMESCAPE review, AMOS (useful routines), Adventures (dead?), ST PD, etc.

Adventure Coder   Issue 23